Assignment2 - Step 4

Phoebe Zhou

September 2025

Parameter Comparison Analysis

In this phase, I wanted to go beyond the basic Naive RAG setup and see how much the system's performance depends on embedding size, retrieval depth, and how we handle multiple retrieved passages. Step 3 already showed that prompt style has a big effect, so in Phase 4 I fixed the prompting and focused on the retrieval side of things.

1. Setup

I tested three embedding models from the MiniLM family:

- MiniLM-L3 (256 dimensions) – smaller and faster, designed for efficiency.
- MiniLM-L6 (384 dimensions) – a slightly larger variant, often a sweet spot between speed and accuracy.
- Multi-qa-MiniLM-L6-cos-v1 (384 dimensions) – tuned for question answering tasks.

For each, I varied the retrieval depth (k=3, 5, 10) and compared three ways of handling the retrieved passages:

1. First: only use the top-ranked passage.
2. Concat: concatenate the top-k passages into a single context.
3. Best_by_len: pick the passage closest in length to the gold answer.2

Performance was measured on the same 918 test questions using Hugging Face's SQuAD metric (Exact Match and F1).

2. Findings

1. <u>Concatenating more passages really helps.</u>

   When I combined the top 10 passages into a single context, F1 jumped above 42 for both MiniLM-L6 and Multi-qa-MiniLM. This beat the top1 baseline by a margin of roughly 3 to 5 points. The extra evidence gave the generator more room to "see" the answer, even if the highest-ranked passage wasn't perfect on its own.

2. Embedding size mattered less than expected.

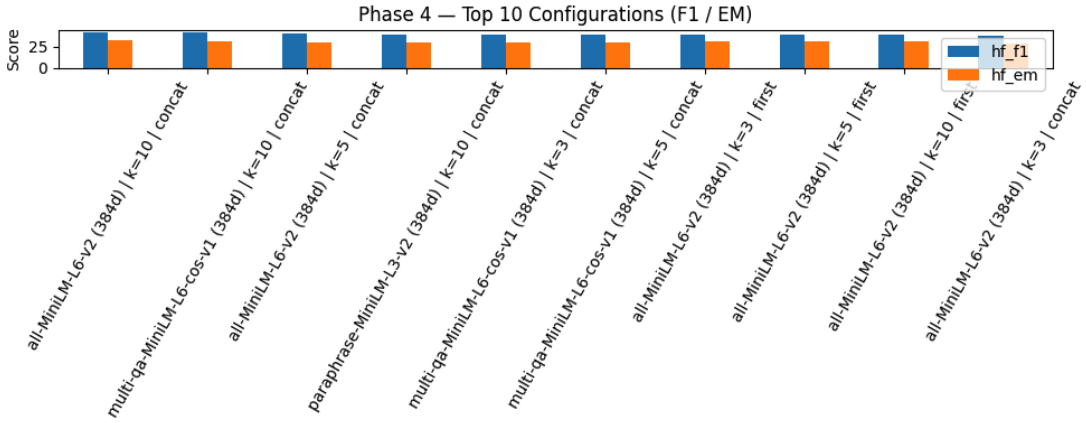
   I thought the 256-dimensional model might trail far behind, but it held its ground. MiniLM-L3 scored F1 at 40 under the best settings, just a couple points behind the 384-dimensional models. That's good news for anyone who needs speed or lighter compute, since smaller embeddings mean less memory and faster indexing.

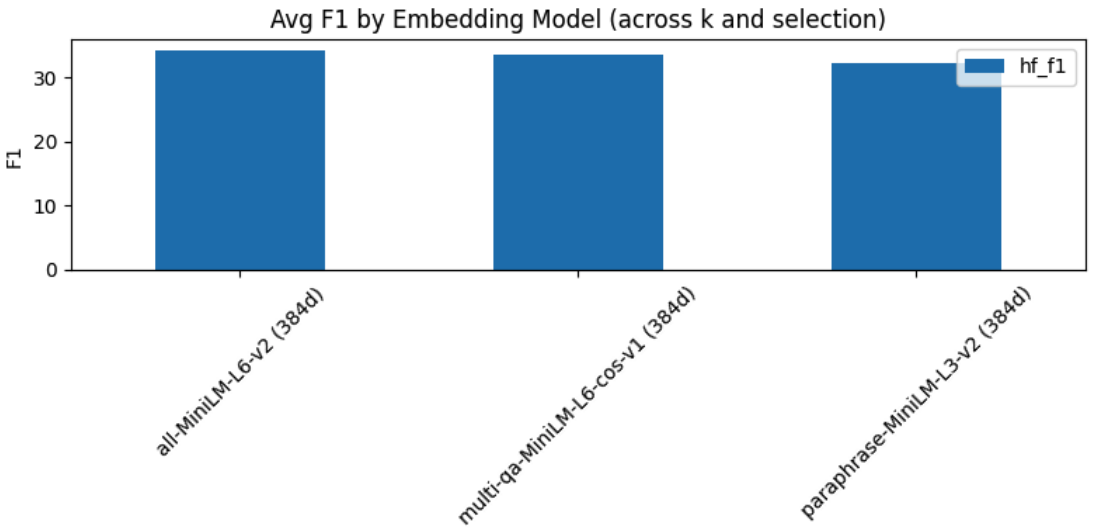
3. Heuristic strategies aren't a substitute for semantics.

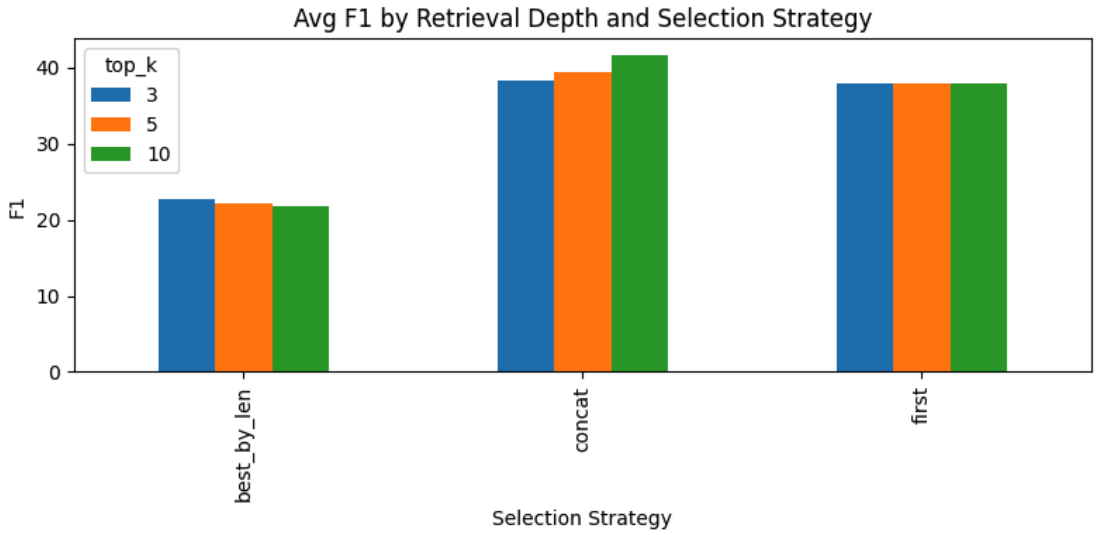
   The "best_by_len" approach performed poorly across the board (F1 stuck around the low 20s). Picking a passage based on length similarity is just not a reliable way to find relevant content. This highlights a bigger point: RAG systems benefit from smarter reranking, not shortcuts.

4. Instruction prompting continues to shine.

   Even though I only used one prompting style here, the consistency of results across models and retrieval depths confirms that Step 3's takeaway holds: clarity and precision in prompts matter as much as the backend retrieval.







3. Reflection

The big lesson from Phase 4 is that retrieval design matters more than raw embedding size. Better embeddings help, but the system's real gains came from giving the generator more high-quality evidence (top-10 concat) instead of just the top-1 passage. At the same time, the small gap between 256d and 384d embeddings suggests there's room to optimize for efficiency without losing too much accuracy.